Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

кафедра програмних засобів

ЗВІТ

з лабораторної роботи №1

з дисципліни «**Спортивне програмування**» на тему:

**«**РЕКУРЕНТНІ ПОСЛІДОВНОСТІ»

Виконав:

ст. гр. КНТ-113сп Артем КЛЕЩОВ

Прийняв:

ст. викл.: Сергій ЛЕОЩЕНКО

2024

# Мета роботи

1.1.1 Вивчити основні можливості та принципи роботи з мовою рекурентні послідовності та співвідношення.

# Завдання до лабораторної роботи

2.1.3 Задано масив M[1:N] натуральних чисел, упорядкований за не спаданням, тобто: M[1] ← M[2] ← ⋯ ← M[N].

2.1.8 Звести число a в натуральну ступінь n за якомога меншу кількість множень.

2.1.9 Задані z та y – дві послідовності. Чи можна отримати послідовність z викреслюванням елементів з y.

# 3. Текст розробленої програми

Задача 3:

function findMinimumNotes(*M*, *S*) {

  let notesCount = [];

  let remainingSum = *S*;

  for (let i = 0; i < *M*.length; i++) {

      const currentNote = *M*[i];

      const count = Math.floor(remainingSum / currentNote);

      if (count > 0) {

          notesCount.push({ note: currentNote, count });

          remainingSum -= count \* currentNote;

      }

  }

  return (!remainingSum) ? notesCount : "Неможливо виплатити задану суму мінімальною кількістю купюр";

}

const M = [200, 100, 50, 20, 10, 5, 2, 1];

const S = 257;

const result = findMinimumNotes(M, S).reverse();

console.log(result);

Задача 8:

function power(*a*, *n*) {

  if (*n* === 0) {

    return 1;

  }

  if (*n* % 2 === 0) {

*// n - парне число*

    const temp = power(*a*, *n* / 2);

    return temp \* temp;

  } else {

*// n - непарне число*

    const temp = power(*a*, (*n* - 1) / 2);

    return *a* \* temp \* temp;

  }

}

let a = 2;

let n = 10;

console.log(`${a} в ступені ${n} = ${power(a, n)}`);

a = 2;

n = 9;

console.log(`${a} в ступені ${n} = ${power(a, n)}`);

Задача 9:

function isSubsequence(*z*, *y*) {

  let zPointer = 0;

  let yPointer = 0;

  while (zPointer < *z*.length && yPointer < *y*.length) {

    if (*z*[zPointer] === *y*[yPointer]) {

      zPointer++;

    }

    yPointer++;

  }

  return zPointer === *z*.length;

}

const z = [1, 3, 5];

const y = [1, 2, 3, 4, 5, 6];

if (isSubsequence(z, y)) {

  console.log(

    'Послідовність z можна отримати з послідовності y викреслюючи елементи.'

  );

} else {

  console.log(

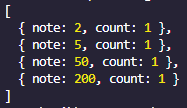
    'Послідовність z НЕ можна отримати з послідовності y викреслюючи елементи.'

  );

}

# 4. Копії екрану, що відображають результати виконання лабораторної роботи.

Задача 3:



Задача 8:



Задача 9:



# 5. Висновки

Під час виконання лабораторної роботи, я вивчив основні можливості та принципи роботи з мовою рекурентні послідовності та співвідношення.